CPSC 413 — Solutions for Quiz #5

Take Home Test on Greedy Methods

November 26, 1998

This take-home test included four different problems, with varying difficulties. Students were asked to choose and solve one of these questions.

1. For the first problem, you were asked to modify the solution for the "optimal fee problem" from the notes on greedy methods, so that it was possible to consider fees that were negative (as well as zero or positive) integers.

   As it happens, two different algorithms for the problem were given — a simple iterative algorithm, which was discussed in the early *lectures,* but which wasn't included in the printed lecture notes, and a recursive algorithm that was in the notes (and discussed briefly in the *later* lectures).

   The following solution will work from the recursive version, just because this will make it easier to use the proof outlines that were also included in the lectures. However, it is possible to solve this problem correctly by modifying the iterative algorithm (and its proof of correctness) instead.

   A very **brief** sketch of the modifications you should have made, to solve this problem by working from the iterative algorithm, will be given at the end.

   (a) As noted in the question, the original greedy algorithm is incorrect when negative fees are allowed, because it *always* returns a valid set of full size k as its output.

   To see that this *would* cause problems, it's sufficient to consider an instance where $n = k = 1$ and $f_1 = -1$.

   The original algorithm would return the set $S = \{1\}$ as output, and this set has total fee $f_S = f_1 = -1$. This is clear, because this set is the only valid set of size one!

   However, the set $S = \emptyset$ (containing no elements at all) is also a "valid" set — it's a subset of $\{1\}$ with size at most one — and its total fee is $f_\emptyset = 0$.

   Since this total fee, 0, is higher than the total fee for the set that is returned by the original algorithm, this proves that the original algorithm would be incorrect if negative fees were allowed.

   Note that this proof would be *exactly* the same if the iterative version of the algorithm was used instead of the recursive one (because we've seen that they both return the same outputs).

   Any proof that gave a *different* instance on which the original algorithm would give the wrong answer would be acceptable, too. This would be an instance such that $1 \leq k \leq n$, and that included n fees $f_1, f_2, \dots, f_n$, such that *strictly fewer* than k of these fees were nonnegative.

(b) It turns out that the *smallest* change you'd need to make to the algorithm and its proof, in order to correct it, is *very small*! All you need to do is modify the definition of "base instances."

Suppose, in particular, that you decided that an instance including integers n and k, and n fees $f_1, f_2, \ldots, f_n$, is a **base instance** if and only if either $k = 0$ **or** $f_i < 0$ for all i such that $1 \le i \le n$, **or both**.

If you decided that the algorithm should return the output set $S = \emptyset$ whenever its instance was a base instance, then it should be clear that the algorithm returns the only possible correct output in this case — the empty set *is* a valid set, and it's the only one whose total fee is nonnegative if the instance is a base instance (either because $k = 0$, so that it's the only set that's valid, or because $f_i < 0$ for all i, so that the total fee of every other valid set would be negative).

Now, suppose that you're given a "non-base" instance of the problem. This now includes integers n and k such that $1 \le k \le n$ (since k is nonzero), and it also includes a sequence of n integer fees $f_1, f_2, \ldots, f_n$ such that $f_i \ge 0$ for at least one integer i (otherwise all the fees would be negative).

Under these circumstances we'll consider a "greedy choice" to be an integer i such that $1 \le i \le n$ and $f_i \ge f_j$ for every integer j such that $1 \le j \le n$. That is, the "greedy choice" is defined just like it was before.

Since this *isn't* a "base instance," $f_i \ge 0$ if i is the greedy choice.

Now, it is possible to produce a "Greedy Choice Property" by modifying the wording of the "claim" that appears on page 76 of Part II of the lecture notes, just to reflect the change in the definition of a "base instance:"

Claim 1.1 (Greedy Choice Property). *If I is an instance of the (generalized) optimal fee problem that includes n jobs, with fees $f_1, f_2, \ldots, f_n$, such that $f_j \ge 0$ for at least one integer j between 1 and n, as well as an integer k such that $1 \le k \le n$, then there exists an integer i such that $f_i \ge f_j$ for all integers j such that $1 \le j \le n$. Furthermore, for any such integer i, there exists an optimal set S of jobs for this instance such that $i \in S$.*

The proof is similar, but not identical, to the proof of the original claim (which you'll find on pages 76–77 of Part II of the notes). Changes to the original proof, in the one that follows, will be shown in bold face.

Proof. Suppose I is as described in the claim (so that it's an instance of the problem that is not a base instance — it includes n jobs with fees as above, for $n \ge 1$, **at least one of these fees is nonnegative**, and it includes the input $k \ge 1$).

Since the set of integers between 1 and n is a nonempty finite set (it is given that $n \ge 1$), **and since one or more of these integers correspond to nonnegative integer fees, with all other fees being negative**, it is clear that there exists at least one integer i such that $1 \le i \le n$ and $f_i \ge f_j$ for all j such that $1 \le j \le n$, as claimed.

Let i be any such integer (so that i is a "greedy choice").

Since there is a finite, but positive, number of subsets of $\{1, 2, \ldots, n\}$ with size at most k, there is at least one such set S such that f_S is maximized. That is, there exists at least one correct output set for this instance.

Now, let $\tilde{S} \subseteq \{1, 2, \ldots, n\}$ be any such correct (valid and optimal) output set corresponding to the instance I.

Either $i \in \tilde{S}$ or $i \notin \tilde{S}$.

Case: $i \in \tilde{S}$. In this case it is sufficient to set S to be $\tilde{S}$; then S is an optimal set of jobs for this instance that includes i, as desired.

Case: $i \notin \tilde{S}$.
Two "subcases" will be considered: Either $|\tilde{S}| < k$ or $|\tilde{S}| = k$. Note that, since $\tilde{S}$ is valid, $|\tilde{S}| \leq k$, so that one of these two subcases must hold.

Subcase: $|\tilde{S}| < k$. In this case, since $|\tilde{S}|$ and k are integers, $|\tilde{S}| \leq k - 1$. Let

$$S = \tilde{S} \cup \{i\}.$$

Clearly $i \in S$ (by construction).
Since $i \notin \tilde{S}$, $|S| = |\tilde{S}| + 1 \leq (k-1) + 1 = k$, and $S \subseteq \{1, 2, \dots, n\}$, so that S is a valid set.
Recall that, since I is not a "base instance," $f_j \geq 0$ for at least one integer j such that $1 \leq j \leq n$. Since i has been chosen such that $f_i \geq f_j$ for all j between 1 and n, this clearly implies that $f_i \geq 0$.
Now, since $i \notin \tilde{S}$ and $f_i \geq 0$,

$$f_S = f_{\tilde{S}} + f_i \geq f_{\tilde{S}}.$$

On the other hand, this is a maximization problem, $\tilde{S}$ is optimal, and S is valid, so that $f_{\tilde{S}} \geq f_S$ as well. Therefore $f_S = f_{\tilde{S}}$ and S is optimal (since $\tilde{S}$ is).

Subcase: $|\tilde{S}| = k$. **The proof for this subcase is identical to the original proof for the same subcase, and can be found on pages 76–77 of Part II of the lecture notes.** $\square$

It turns out that the constructions that were given for the original algorithm (to deal with the "optimal substructure property") and the proofs that these constructions are correct remain valid even if negative fees are allowed! Therefore, you could define exactly the same constructions and give exactly the same proof as you in Section 3.7.2, on pages 77–79 of Part II of the notes, in order to prove the "optimal substructure property."
The corresponding recursive solution for this problem would be identical to the algorithm shown in Figure 3.1 on page 61 of the notes except that the very first line would involve a more complicated test: It should now be something like

$$\textbf{if } (k = 0 \textbf{ or } n = 0 \textbf{ or } f_j < 0 \text{ for all } j) \textbf{ then}$$

in order to reflect the change in the definition of a "base instance."
Since the greedy choice and optimal substructure properties have been proved, one could complete the proof of correctness of this recursive algorithm using a proof by induction on the "size" n of the input. However, this kind of proof wasn't expected as part of an answer for this question.

(c) *Alternative Solutions:* When this question was set, I had a slightly different solution in mind — namely one in which "sentinel values" were used in some way in order to prevent a fee f_i from being included more than once in a solution.

In particular, suppose that the notion of the "size" of an instance is changed, to be the value k of the last part of the input. It turns out that you could then replace the first "construction" for the optimal substructure property — in which you produce a smaller instance of the same problem, to be solved recursively — with a construction in which you start with an instance $n, f_1, f_2, \dots, f_n$, and k, and a greedy choice i, and produce a "smaller" instance, $\hat{n}, \hat{f}_1, \hat{f}_2, \dots, \hat{f}_n$, and $\hat{k}$, by setting $\hat{n} = n$, (**not** $n-1$!), $\hat{k} = k - 1$, and, for $1 \le j \le n$,

$$\hat{f}_j = \begin{cases} f_j & \text{if } j \neq i, \\ -1 & \text{if } j = i. \end{cases}$$

You would then replace the *second* construction by one in which you would take a solution $\hat{S} \subseteq \{1, 2, \dots, n\}$ for the above "smaller" instance, and produce a solution S for the original instance, by setting

$$S = S \cup \{i\}$$

in all cases!

In order to prove correctness of *this* pair of constructions, you'd need to establish a separate result:

Lemma 1.2. *Suppose S is a solution for an instance n, $f_1, f_2, \dots, f_n$, k of the generalized optimal fee problem, and $i \in S$. Then $f_i \ge 0$.*

Proof. Suppose, instead, that $i \in S$ and $f_i < 0$. Then if

$$\tilde{S} = S \setminus \{i\},$$

then $\tilde{S} \subseteq \{1, 2, \dots, n\}$ (since $S \subseteq \{1, 2, \dots, n\}$) and $|\tilde{S}| \le |S| \le k$, so that $\tilde{S}$ is valid. Furthermore, since $S = \tilde{S} \cup \{i\}$ and $i \notin \tilde{S}$,

$$f_S = f_{\tilde{S}} + f_i < f_{\tilde{S}}$$

if $f_i < 0$, which contradicts the fact that S is optimal. □

This "lemma" implies that if $\hat{S}$ really *is* a solution for the smaller instance defined by the new construction then $i \notin \hat{S}$, since $\hat{f}_i = -1 < 0$.

Once this is established, the optimal substructure property can be established by **simplifying** the argument given in Section 3.7.2 of Part II of the notes.

Since this alternative required a bit more work but also leads to a simpler (and therefore, somewhat better) algorithm, a few *more* than twenty marks will be given for a complete and correct presentation of this version of the algorithm and proof of its correctness.

Here's a slightly different "alternate solution:" At least one student apparently noticed that it would also be sufficient to replace k by the value

$$\hat{k} = \min(k, |\{i \ : \ 1 \le i \le n \text{ and } f_i \ge 0\}|)$$

(which is the minimum of k and the number of fees that are nonnegative), and then execute exactly the same algorithm as before (that is, the algorithm in the notes!), using $\hat{k}$ instead of k, in order to correctly solve this modified version of the problem.

However, the fact that this *does* give a correct solution requires a proof!

In order to prove this, you'd likely need to prove Lemma 1.2, above, and then deduce that this means that the output set must have size $\hat{k}$, as above.

This change — replacing k with the above value $\hat{k}$ in statements of claims and their proofs — is probably the most significant part of what you'd need to do in order to prove correctness of an *iterative* algorithm for the generalized version of this problem, as well.

2. Before considering a greedy algorithm for this problem, please note that the following definition of a "connecting set" for a tree is **equivalent** to the definition given in the problem: A subset C of the nodes in a (rooted) tree T is a connecting set of T if every node v is either a member of C or is *at most one edge away from a member of* C in the tree.

   After all, the nodes that are "at most one edge away from v" are just the parent of v, the children of v, and v itself.

   This alternative definition (which, unfortunately, I only thought of when preparing these solutions) will be used because it *shortens* some claims in the argument — however, you could give pretty much the same algorithm and proof using the original definition instead! (You'd just have to write a bit more.)

   In order to solve this problem, we'll begin by defining **base instances** and their solutions.

   We'll consider an instance of this problem — that is, a rooted tree T — to be a "base instance" if there is at most one node in T.

   It should be clear that it's easy to *recognize* the base instances, because the *only* base instances are empty trees (with no nodes at all) and trees of size one (consisting only of a root).

   If a base instance T is an "empty tree" — it has no nodes at all — then the only connecting set for this tree is the empty set. Thus, the solution, for this kind of base instance, is the empty set, $\emptyset$.

   If a base instance T is a tree of size one — including only a root node r — then the only connecting set for this is the set $\{r\}$ that includes this root, so that a solution for this instance is the set $\{r\}$ in this case.

   Suppose now that we have a "non-base instance" — a tree T that includes at least two nodes.

   We'll define a *greedy choice* for this instance to be any node v that is the *parent* of one (or more) of the deepest leaves in the tree.

   Note that this is **exactly the same** as the "greedy choice" that was used for the "vertex cover" problem (on trees) discussed in the labs.

   Of course the corresponding "greedy choice property" and its proof will be at least a little bit different!

   Claim 2.1 (Greedy Choice Property). *Suppose T is a tree that includes at least two nodes.*

   Then there exists a node v in T which is a parent of one (or more) of the deepest *leaves in the tree.*

   Now let v be any node in T that is *the parent of one (or more) of the deepest leaves in the tree. Then there exists a connecting set C for T, such that $|C| \leq |C'|$ for every other connecting set C' for T, and such that $v \in C$.*

Proof. Suppose, as above, that T is a tree with at least two nodes.

Since T is nonempty it includes at least one leaf, u. Since it only has *finitely many* leaves, there is at one leaf, w, that is at least as deep in the tree as all the other leaves.

Since T includes at least two nodes, the root is not a leaf. Therefore w is not the root, and w has a parent. Clearly, if v is the parent of w, then v is "a parent of one (or more) of the deepest leaves in the tree."

Thus, every tree with two or more nodes *has* a node v that's the parent of one (or more) of the deepest leaves in the tree, as claimed.

Note as well that there is at least one "connecting set" for T — namely, the set of all the nodes in the tree. There are only finitely many (but, more than zero) connecting sets, since each connecting set is a subset of the nodes in T and T only has finitely many nodes.

Since the size of a connecting set is finite and is a nonnegative integer, it should be clear that there is at least one connecting set whose size is less than or equal to the sizes of all the others.

In other words, if T is an instance of the problem defined in this question, then this instance *does* have at least one solution — an optimal subset of the nodes of T does exist.

Now, let v be a node in T that is a "parent of one (or more) of the deepest leaves in the tree," and let $\tilde{C}$ be connecting set for T whose size is as small as possible, so that $\tilde{C}$ is an optimal set of the nodes in T.

As usual, two cases are possible: either $v \in \tilde{C}$ or $v \notin \tilde{C}$.

If $v \in \tilde{C}$ then it is sufficient to set C to be $\tilde{C}$: C will then be a connecting set of minimal size that includes v, as required.

Suppose instead that $v \notin \tilde{C}$. Suppose, as above, that w is one of the deepest leaves in the tree whose parent is v.

Since $v \notin \tilde{C}$, and the only nodes that are at most one edge away from w are v and w itself, $w \in \tilde{C}$ — otherwise $\tilde{C}$ would not be a connecting set!

Now let

$$C = (\tilde{C} \setminus \{w\}) \cup \{v\}$$

so that C includes the same nodes as $\tilde{C}$ except that w has been removed and v has been included.

Clearly C is a subset of the nodes of the tree, since $\tilde{C}$ is.

Now let u be any node in the tree. Either

(a) u is one of the children of v,

(b) u is the node v itself, or

(c) u does not belong to the subtree that has v as its root at all.

These are only possible cases because v is a parent of one of the *deepest* nodes in the tree — all the children of v must be leaves.

In case (a), u is one edge away from v, and $v \in C$.

In case (b), u is the same node as v and $v \in C$, so $u \in C$.

In the final case (c), u is at most one edge away from some node $x \in \tilde{C}$, since $\tilde{C}$ is a connecting set. Since u is not in the subtree whose root is v at all, u is strictly *more* than just one edge away from w — so x is a different node than w.

This implies (by the construction of C from $\tilde{C}$) that $x \in C$ as well, so u is at most one edge away from some element of C.

Since *every* node of the tree is at most one edge away from some element of C, C is a connecting set.

Therefore C is a "valid" subset of the nodes of the input tree.

Finally, since $w \in \tilde{C}$ and $v \notin \tilde{C}$, it should be clear (from the construction of C from $\tilde{C}$) that $|C| = (|\tilde{C}| - 1) + 1 = |\tilde{C}|$. Since $\tilde{C}$ is a connecting set of minimal size, C is a connecting set with minimal size as well — the sizes of the two sets are the same!

Therefore C is also an optimal subset of the nodes of the tree, since $\tilde{C}$ is.

Since $v \in C$ (by construction), C has all the properties needed to complete the proof. □

Now, in order to prove the "optimal substructure property," it is necessary to present two constructions and prove that each is correct.

The first construction should take as input

- any "non-base instance" of this problem — that is, any tree T with two or more nodes, and
- any "greedy choice" for this instance — that is, any parent v of one (or more) of the deepest leaves in T,

and it should produce a strictly *smaller* instance of the same problem as output — that is, another tree $\hat{T}$ with strictly fewer nodes than T.

There are three cases that must be considered, and there is a different way to produce $\hat{T}$ for each.

In the first case, v is the root of T. In this case, $\hat{T}$ should be the empty tree (with zero nodes).

In the second and third cases, v is not the root; let x denote the parent of v in each case.

In the second case, x has only one child — which must be v. In this case, the tree $\hat{T}$ should include all the nodes that T does *except for* x, v, and all the children of v. Note that, since v was a parent of one or more of the deepest leaves in the tree, v doesn't have any "grandchildren." Thus, $\hat{T}$ includes all the nodes of T except for the nodes in the subtree with x as root.

In the third case, x has two or more children. In this case, the tree $\hat{T}$ should include all the nodes that T does *except* for v and its children. Since v has no "grandchildren," $\hat{T}$ includes all the nodes of T except for the nodes in the subtree with v as its root.

Therefore, this construction is different from the one used in the above second case, because x is included in $\hat{T}$ in the third case, but not the second.

Before giving the second construction, let's consider the correctness of the first one.

Claim 2.2. *If T and v are as above and $\hat{T}$ is the tree (or "forest") that is described as above, then $\hat{T}$ is a tree which includes a strictly smaller number of nodes than T.*

Proof. First note that $\hat{T}$ includes a subset of the nodes of T.

Note as well that $\hat{T}$ is either the empty tree, or is obtained from T by removing the *subtree* of T with some node (either x or v) as its root in every case. As a result, $\hat{T}$ is also a tree in every case.

Finally, $\hat{T}$ never includes the greedy choice, v. Therefore $\hat{T}$ really is a tree with strictly fewer nodes than T, as required. □

Now let's consider the second construction. This should take as its input

- any "non-base instance" of the problem — that is, any tree T with two or more nodes,
- any "greedy choice" for this instance — that is, any parent v of one (or more) of the deepest leaves in T,
- the smaller instance $\hat{T}$ of the problem (a smaller tree) that would be produced from T and v using the first construction, and
- any solution for the smaller instance — that is, a connecting set $\hat{C}$ of $\hat{T}$ whose size is as small as possible,

and it should produce a solution for the original problem — a connecting set C of T whose size is as small as possible — as output.

It turns out that the second construction is extremely simple: It just defines the set C to be

$$C = \hat{C} \cup \{v\},$$

that is, the set obtained from $\hat{C}$ by including the greedy choice.

The following claim implies that this second construction is also correct.

Claim 2.3. *Let T be a tree with two or more nodes, let v be a parent of one or more of the deepest leaves in T, and let $\hat{T}$ be the smaller tree produced using the first construction from T and v. Let $\hat{C}$ be any connecting set for $\hat{T}$ of minimal size. Then if*

$$C = \hat{C} \cup \{v\},$$

then C is a connecting set for T of minimal size. That is, C is an "optimal" subset of the nodes of T.

Proof. Obviously, C is a subset of the nodes of T (it has "the right type").

We'll continue by proving that C is "valid" — that is, that it is a connecting set of T.

Recall that $\hat{T}$ could have been produced from T and v in one of three ways, depending on whether

- v was the root of T,
- v had a parent, w, in T, and w had only one child (which was clearly just the node v), or
- the parent w of v had two or more children in T.

In the first case (v was the root of T), the fact that v is also a parent of one of the *deepest* leaves of T implies that the *only* nodes in T were v and its children.

Since the first construction returned the empty tree as $\hat{T}$ in this case, and $\hat{C}$ was also the empty set, $C = \{v\}$ in this case. Since the only nodes in T *besides* v are all children of v, they are all "at most one edge away" from v in T. Thus the set $C = \{v\}$ is a connecting set in this case.

In the second case (the parent w of v only had one child — v), $\hat{T}$ included all the nodes of T except for w, v, and the children of v. All of these nodes (w, v, and v's children) are "at most one edge" away from v so, since $v \in C$, they're at most one edge away from some element of C. All of the rest of the nodes belong to $\hat{T}$, and $\hat{C}$ is a connecting set for $\hat{T}$, so they're all at most one edge away from some member of $\hat{C}$. Since $\hat{C} \subseteq C$, they're all at most one edge away from a member of C as well. Therefore, every node in T is at most one edge away from a member of C — and C is a connecting set for T in this case too.

The proof that C is a connecting set in the third case is almost identical to the proof in the second case (the only difference will be that it will turn out that w is at most one edge away from at least *two* members of C — which won't prevent C from being a connecting set).

Thus C is always a connecting set — it's a "valid" subset of the nodes of T.

All that's left to do is to prove that C is also optimal. Unfortunately, it seems to be necessary to give a slightly different argument for each of the three cases (based on how $\hat{T}$ was constructed from T) here as well.

In the first case — v was the root of T, so that $\hat{T}$ is the empty tree — it's been noted already that $C = \{v\}$. Since T *has* at least one node, the empty set is not a connecting set for T, so that any connecting set of T with size one — including the above set C — must be optimal.

Now consider the second case — the parent w of v had exactly one child.

If w was the root of T then, once again, $\hat{T}$ must have been the empty tree — and, once again $C = \{v\}$. Once again, the empty set is not a connecting set, so C is clearly optimal.

Suppose, therefore, that w is not the root of T, so that w has a parent, "x," in T. Note that x is one of the nodes in $\hat{T}$ (by inspection of the first construction).

It follows by the greedy choice property that there is some connecting set D for T of minimal size, such that $v \in D$.

It can be shown, without too much effort, that if $w \in D$ then the set

$$(D \setminus \{w\}) \cup \{x\}$$

is also a connecting set of T of minimal size, which includes v, and — now — *does not* include w. (If this is not clear, please prove this as an exercise.)

Therefore we may assume without loss of generality that D is a connecting set of minimal size for T such that $v \in D$ and $w \notin D$. Since D *does* have minimal size, D does not include any of the children of v, either — if it did then (since $v \in D$ and all the children of v are leaves) we could remove these children from D and produce an even smaller connecting set for T.

Thus, if

$$\hat{D} = D \setminus \{v\}$$

then $\hat{D}$ *is a subset of the set of nodes in* $\hat{T}$.

Furthermore, since all the node in $\hat{T}$ are at least *two* edges away from v in T, and D is a connecting set for T, $\hat{D}$ is a connecting set for $\hat{T}$.

To see this, let z be some node in $\hat{T}$. Then z is a node in T as well. Since D is a connecting set for T, z is at most one edge away from some node $a \in D$. This node a must be different from v (because every node in $\hat{T}$ is at least two edges away from v), so $a \in \hat{D}$ as well. Thus z is also at most one edge away from an element of $\hat{D}$ and, since z was arbitrarily chosen, this implies that $\hat{D}$ is a connecting set for $\hat{T}$.

Therefore

$$|\hat{C}| \leq |\hat{D}|$$

since $\hat{C}$ was a connecting set for $\hat{T}$ with minimal size.

It should be clear that $|C| = |\hat{C}| + 1$ and $|D| = |\hat{D}| + 1$, so this implies that

$$|C| \leq |D|$$

as well. On the other hand, C is a connecting set for T and D is a connecting set for T with minimal size, so

$$|D| \leq |C|$$

too. Therefore

$$|C| = |D|,$$

and C is a connecting set for T with minimal size, since D is.

That is, C is an "optimal" subset of the nodes of T in this second case.

Finally, let's consider the third case — the parent w of v had two or more children in T. In this case, $\hat{T}$ was obtained from T by removing v and all its children — so that w is included in $\hat{T}$.

As before, we will continue by noting that the greedy choice property implies that there is some connecting set D for T of minimal size, such that $v \in D$.

We may assume without loss of generality that D does include any of the leaves of T — for, if $t \in D$ and t is a leaf, then we could replace t in D with its parent. It would not be difficult to prove that the resulting set is still a connecting set if D was and that it included one fewer leaf. The size of the set would clearly not have increased and it would still contain v (since v is not a leaf). If you did this for every leaf that *did* belong to D you'd have a connecting set of minimal size, including v and not including any of the leaves, as desired.

Recall that the parent w of v has at least two children. Since v is a parent of one or more of the deepest leaves in T, w cannot possibly have any "great-grandchildren:" It is the root of a subtree with depth two in T.

Now, since D is a connecting set for T that doesn't include any leaves, and w has at least one child besides v, D must include either

- one or more of the children of w *besides* v — because these children are parents of a set of leaves (which must somehow be "at most one edge away" from an element of D), or

- w itself (because one or more of w's other children is a leaf),

or both.

Let

$$\hat{D} = D \setminus \{v\}.$$

Since all of the children of v are leaves and D doesn't include any of these, $\hat{D}$ is a subset of the set of nodes in $\hat{T}$. (Please look again at the first construction if it isn't clear that this is true.)

Suppose that b is a node in $\hat{T}$. If b is the parent w of v then b is at most one edge away from some element of $\hat{D}$: Note that, as claimed above, $\hat{D}$ includes either w itself or at least one of w's children (that isn't v).

If b is a node in $\hat{T}$ different from w then there is some node c in D such that b is at most one edge away from c. Since b is different from w, and $b \in \hat{T}$, b is at least two edges away from v. Therefore, c is some element of D that is different than v, so that c is in $\hat{D}$. Therefore b is also at most one edge away from some element of $\hat{D}$.

Therefore, $\hat{D}$ is a connecting set for $\hat{T}$.

Now the rest of the proof is exactly the same as it was for the second case was, after it was concluded that $\hat{D}$ was a connecting set for $\hat{T}$: One simply observes that $|\hat{C}| \leq |\hat{D}|$, notes that this implies that $|C| \leq |D|$ as well, establishes (as above) that C and D must then have the same size, and then concludes that C has minimal size, since D does.

Therefore C is a connecting set for T with minimal size, as desired, in this last case as well. □

Now that we have a way to recognize "base instances" and solve them, a greedy choice, and the two constructions needed for the "optimal substructure property," a recursive greedy algorithm for this problem can be obtained by plugging these components into the generic "greedy method" shown in Figure 3.3 on page 65 of Part II of the lecture notes.

The correctness of this algorithm could be established (using the correctness of the method to deal with base instances, and the greedy choice and optimal substructure properties) by induction on the number of nodes in the input tree. This proof by induction was not required for a complete solution for this question, if everything else had been done.

Finally, the proof of correctness of the second construction for the "optimal substructure property" is more complicated than I'd anticipated. Therefore it will be possible to earn slightly more than 25 marks for a complete and correct solution of this problem — and, therefore, it will be possible to receive full marks (or high marks) even if you missed one or two of the subcases or subtle parts of the proof given here.

3. It's likely that the trickiest part of this question concerns "validity" rather than "optimality" of solutions.

   (a) Let's begin (as you'd expect) by considering the new "minimization" problem, in which you still wish to make change using as few coins as possible.

   The following lemma will be useful later on.

Lemma 3.1. *Suppose $n > 0$, and let c, k, $m_0, m_1, \ldots, m_k$ and n be* any *instance of this new coin-changing problem including n.*

Suppose that it is possible to make change for n using the available coins — and let $s_0, s_1, \ldots, s_k$ be any *nonnegative integers such that*

$$0 \leq s_i \leq m_i$$

for all i (such that $0 \leq i \leq k$) and

$$\sum_{i=0}^{k} s_i c^i = n.$$

Then there exists at least one integer i such that $0 \leq i \leq k$, $c^i \leq n$, and $m_i > 0$.

Furthermore, if l is the largest integer such that $0 \leq l \leq k$, $c^l \geq n$, and $m_l > 0$ (so that either $c^i > n$ or $m_i = 0$ or both, for all i such that $l + 1 \leq i \leq k$), then either $s_l > 0$ or there exists a set of nonnegative integers $t_0, t_1, \ldots, t_{l-1}$ such that

$$0 \leq t_i \leq s_i \leq m_i$$

for $0 \leq i \leq l - 1$, and such that

$$\sum_{i=0}^{l-1} t_i c^i = c^l.$$

Proof. First, it should be noted that since it's possible to make change for n, there *is* some integer i such that $0 \leq i \leq k$, $c^i \leq n$, and $m_i > 0$ — for otherwise, you couldn't possibly make change for n because all the available coins would be worth too much.

The rest of the proof will use induction on n.

Basis: The claim is trivial if $n = 0$ (since it requires that n is positive). Suppose, therefore, that $n = 1$. In this case, since it is possible to make change for n, it must be true that $s_0 = 1$, $m_0 \geq 1$, $s_i = 0$ for $1 \leq i \leq k$, and $l = 0$ (since $1 = n < c$). Therefore $s_l = s_0 = 1$, and this is all that's needed.

Inductive Step: Suppose $\hat{n} > 0$ and that the result is true for every integer less than or equal to $\hat{n}$.

Now let $n = \hat{n}+1$ and consider an instance of this problem including c, k, $m_0, m_1, \ldots, m_k$, and n such that it's possible to make change for n. As argued above, there is at least one integer i such that $0 \leq i \leq k$, $c^i \leq n$, and $m_i > 0$, so (since every such integer is less than or equal to k) there is also some such *maximal* integer l; let l and $s_0, s_1, \ldots, s_k$ be as in the statement of the lemma.

If $s_l > 0$ then we're done. Suppose, therefore, that $s_l = 0$.

Since either $c^i > n$ or $m_i = 0$, when $l + 1 \leq i \leq k$, the s_i's are all nonnegative integers, and

$$\sum_{i=0}^{k} s_i = n,$$

it is clear that $s_i = 0$ when $l+1 \leq i \leq k$ (otherwise the above sum would be too large, or we'd have $s_i > m_i$ for one or more i). Therefore, since $s_l = 0$ as well,

$$\sum_{i=0}^{l-1} s_i = n.$$

Obviously, since $n > 0$, this implies that $l > 0$.

We'll consider three cases: Either $n = c^l$, $n > c^l$ and $s_0 = 0$, or $n > c^l$ and $s_0 > 0$.

In the first case, $n = c^l$, we're done — we can set $t_i = s_i$ for $0 \leq i \leq l-1$. Then $0 \leq t_i \leq s_i \leq m_i$ for $0 \leq i \leq l-1$ and

$$\sum_{i=0}^{l-1} t_i c^i = \sum_{i=0}^{l-1} s_i c^i = n = c^l$$

as required.

If $n > c^l$ and $s_0 = 0$ then, since $n > 0$ and

$$\sum_{i=1}^{l-1} s_i c^i = \sum_{i=0}^{l-1} s_i c^i = n,$$

it is clear that $n \geq c$ and, furthermore, that n is divisible by c (after all, every term in the sum on the left is divisible by c).

Now let $\overline{n} = n/c$, so $1 \leq \overline{n} \leq \hat{n} = n-1$, and let $\overline{c} = c \geq 2$, $\overline{k} = k-1$, let $\overline{m}_i = m_{i+1}$ for $0 \leq i \leq \overline{k} = k-1$, and let $\overline{s}_i = s_{i+1}$ for $0 \leq i \leq \overline{k}$ as well. Then it is easily checked that $\overline{l} = l-1$ is the largest integer such that $0 \leq \overline{l} \leq \overline{k}$, such that $\overline{c}^{\overline{l}} \leq \overline{n}$ (since $\overline{c}^{\overline{l}} = c^{l-1}$ and $\overline{n} = n/c$), and such that $\overline{m}_{\overline{l}} > 0$ (since $\overline{m}_{\overline{l}} = m_{\overline{l}+1} = m_l$). Furthermore it's easily confirmed that $0 \leq \overline{s}_i \leq \overline{m}_i$ for $0 \leq i \leq \overline{k}$ (since $0 \leq s_i \leq m_i$ for $1 \leq i \leq k$) and, since $s_0 = 0$,

$$\begin{aligned}
\sum_{i=0}^{\overline{k}} \overline{s}_i \overline{c}^i &= \sum_{i=0}^{k-1} s_{i+1} c^i \\
&= \frac{1}{c} \sum_{i=0}^{k-1} s_{i+1} c^{i+1} \\
&= \frac{1}{c} \sum_{j=0}^{k} s_j c^j \\
&= \frac{n}{c} = \overline{n}.
\end{aligned}$$

It now follows by the inductive hypothesis that either $\overline{s}_{\overline{l}} > 0$ or there exist integers $\overline{t}_0, \overline{t}_1, \ldots, \overline{t}_{\overline{l}-1}$ such that $0 \leq \overline{t}_i \leq \overline{s}_i \leq \overline{m}_i$ for $0 \leq i \leq \overline{l}-1$ and

$$\sum_{i=0}^{\overline{l}-1} \overline{t}_i \overline{c}^i = \overline{c}^{\overline{l}} = c^{l-1}.$$

Since $\overline{s}_{\overline{l}} = s_{\overline{l}+1} = s_l = 0$, the first case is impossible, so the above values $\overline{t}_0, \overline{t}_1, \ldots, \overline{t}_{\overline{l}-1}$ must exist.

Now let $t_0 = 0$ and let $t_{i+1} = \overline{t}_i$ for $0 \le i \le \overline{l}$; this defines a sequence of nonnegative integers $t_0, t_1, \ldots, t_l$. It is easily checked that $0 \le t_i \le s_i \le m_i$ for $0 \le i \le l-1$ and (since $t_0 = 0$),

$$\begin{aligned}\sum_{i=0}^{l-1} t_i c^i &= \sum_{i=1}^{l-1} t_i c^i \\ &= c \sum_{i=1}^{l-1} t_i c^{i-1} \\ &= c \sum_{i=1}^{l-1} \overline{t}_{i-1} c^{i-1} \\ &= c \sum_{j=0}^{\overline{l}-1} \overline{t}_j \overline{c}^j \\ &= c \cdot \overline{c}^{\overline{l}} \\ &= c \cdot c^{l-1} = c^l,\end{aligned}$$

as required.

In the final case, $n > c^l$ and $s_0 > 0$, it suffices to let $\overline{n} = n - 1 = \hat{n}$ and to consider the instance including values $\overline{c} = c$, $\overline{k} = k$, $\overline{m}_0 = m_0 - 1 \ge 0$, and $\overline{m}_i = m_i$ for $1 \le i \le k$, with $\overline{s}_0 = s_0 - 1 \ge 0$ and $\overline{s}_i = s_i$ for $1 \le i \le k$, in order to obtain another case where the inductive hypothesis can be applied. This time it will turn out that $\overline{l} = l$. Once again, it can (eventually) included that there must be a sequence $\overline{t}_0, \overline{t}_1, \ldots, \overline{t}_{\overline{l}-1}$ of nonnegative integers such that

$$\sum_{i=0}^{\overline{l}-1} \overline{t}_i \overline{c}^i = \overline{c}^{\overline{l}} = c^l.$$

Now it suffices to set $t_i = \overline{t}_i$ for $0 \le i \le l-1$ — clearly $0 \le t_i \le s_i \le m_i$ for $0 \le i \le l-1$ and

$$\sum_{i=0}^{l} t_i c^i = c^l$$

in this case — as required to complete the proof. $\square$

Now we're ready to define a greedy algorithm to solve this problem.

Base instances will be those such that $n = 0$. Clearly the only correct solution for any such instance is the sequence $s_0, s_1, \ldots, s_k$ such that $s_0 = s_1 = \cdots = s_k = 0$.

We'll consider the "size" of an instance to be n, so that base instances are the instances with size 0.

The *greedy choice* for any non-base instance is a coin with denomination c^l, where $l = 0$ if either $c^i > n$ or $m_i = 0$ (or both) for all i such that $0 \le i \le k$, and where l is the largest integer such that $c^l \le n$ and $m_l > 0$ otherwise.

The following greedy choice property is stronger a bit different than usual — it establishes that if it possible to make change for n at all, then *every* solution includes at least one coin with denomination c^l.

Claim 3.2. *Let $c \geq 2$, $k \geq 0$, $m_0, m_1, \ldots, m_k \geq 0$ and let $n > 0$ be a non-base instance of this coin changing problem. Furthermore let l be an integer such that either $l = 0$, when $c^i > n$ or $m_i = 0$ for all i such that $0 \leq i \leq n$, and such that l is the largest integer such that $c^l \leq n$ and $m_l > 0$ otherwise.*
Suppose it is possible to make change for n with the available coins. Then $c^l \leq n$ and $m_l > 0$. Furthermore, if $s_0, s_1, \ldots, s_k$ is any *solution for the given instance of the problem, then $s_l > 0$.*

Proof. Suppose it is possible to make change for n; then Lemma 3.1 implies the existence of an integer i such that $0 \leq i \leq k$, $c^i \leq n$, and $m_i > 0$ — and *this* implies that if l is the greedy choice then $0 \leq l \leq k$, $c^l \leq n$, and $m_l > 0$ as well.
Now let $s_0, s_1, \ldots, s_k$ be any solution for this instance of the problem. If $s_l > 0$ then we're done; suppose, therefore, that $s_l = 0$. Now, it's necessary and sufficient to obtain a contradiction (proving that this case is impossible) in order to complete the proof.
Since either $m_i = 0$ or $c^i > n$, so that $s_i = 0$ for every integer $i > l$, the fact that

$$\sum_{i=0}^{k} s_i c^i = n$$

and that $s_l = 0$ implies that

$$\sum_{i=0}^{l-1} s_i c^i = n$$

as well. Now Lemma 3.1 implies that there exist nonnegative integers $t_0, t_1, \ldots, t_{l-1}$ such that $0 \leq t_i \leq s_i \leq m_i$ for $0 \leq i \leq l-1$, and such that

$$\sum_{i=0}^{l-1} t_i c^i = c^l.$$

Consider a sequence $\hat{s}_0, \hat{s}_1, \ldots, \hat{s}_k$ of integers such that, for $0 \leq i \leq l$,

$$\hat{s}_i = \begin{cases} s_i - t_i & \text{if } 0 \leq i \leq l-1, \\ 1 & \text{if } i = l, \\ s_i & \text{if } l+1 \leq i \leq k. \end{cases}$$

Since $t_i \leq s_i$ for $0 \leq i \leq l-1$, and $s_0, s_1, \ldots, s_k$ is a "valid" sequence, $\hat{s}_i \geq 0$ for all i as well. Furthermore

$$\begin{aligned} \sum_{i=0}^{k} \hat{s}_i c^i &= \sum_{i=0}^{l-1} \hat{s}_i c^i + \hat{s}_l c^l + \sum_{i=l+1}^{k} \hat{s}_i c^i \\ &= \sum_{i=0}^{l-1} (s_i - t_i) c^i + 1 \cdot c^l + \sum_{i=l+1}^{k} s_i c^i \\ &= \left(\sum_{i=0}^{l-1} s_i c^i - \sum_{i=0}^{l-1} t_i c^i \right) + c^l + 0 \\ &= (n - c^l) + c^l = n. \end{aligned}$$

Thus the sequence $\hat{s}_0, \hat{s}_1, \dots, \hat{s}_l$ is also "valid."
Finally, it should be noted that

$$\sum_{i=0}^{l-1} t_i \geq 2;$$

for, otherwise,

$$\sum_{i=0}^{l-1} t_i c^i \leq \sum_{i=0}^{l-2} 0 \cdot c^i + 1 \cdot c^{l-1} = c^{l-1} < c^l,$$

contradicting the choice of the t_i's. However, *this* implies that (since $\hat{s}_l = 1$ and $s_l = 0$)

$$\begin{aligned}
\sum_{i=0}^{k} \hat{s}_i &= \sum_{i=0}^{l-1} \hat{s}_i + \hat{s}_l + \sum_{i=l+1}^{k} \hat{s}_i \\
&= \sum_{i=0}^{l-1} (s_i - t_i) + (s_l + 1) + \sum_{i=l+1}^{k} s_i \\
&= \sum_{i=0}^{k} s_i - \sum_{i=0}^{l-1} t_i + 1 \\
&\leq \sum_{i=0}^{k} s_i - 2 + 1 \\
&< \sum_{i=0}^{k} s_i,
\end{aligned}$$

contradicting the fact that $s_0, s_1, \dots, s_k$ was "optimal" as well as valid.
Thus, we've obtained the desired contradiction: s_l must be positive if $s_0, s_1, \dots, s_k$ is a solution for the given instance. ☐

The next claim can now be established from the previous one by induction on n:

Claim 3.3. *Let c, k, $m_0, m_1, \dots, m_k$ and n be an instance of this problem such that it is possible to make change for n, and let $s_0, s_1, \dots, s_k$ be any solution for this instance of the problem. Then*

$$s_k = \min(\lceil \tfrac{n}{c^k} \rceil, m_k).$$

Proof. As noted above, the proof uses (the strong form of) induction on n.

Basis: If $n = 0$ then clearly the only solution of the problem is the sequence $s_0, s_1, \dots, s_k$ such that $s_i = 0$ for all i. Since

$$\min(\lceil \tfrac{n}{c^k} \rceil, m_k) = \min(0, m_k) = 0,$$

the desired identity is satisfied.

Inductive Step: Suppose $n > 0$ that the result is correct for all smaller integers.

If $n < c^k$ then, the same argument as the one used for the basis implies that the result is correct. Suppose, therefore, that $n \geq c^k$.

If $m_k = 0$, then obviously any solution for the problem must have $s_k = 0$ as well, and

$$\min(\lceil \tfrac{n}{c^k} \rceil, m_k) = \min(\lceil \tfrac{n}{c^k} \rceil, 0) = 0,$$

so the desired identity is satisfied in this case, too.

Suppose, therefore, that $n \geq c^k$ and $m_k > 0$. Then the "greedy choice" l mentioned in Claim 3.2 is actually the integer k — and this claim implies that $s_k > 0$ in any solution for this instance of the problem.

Now consider a modified instance in which m_k is replaced with $m_k - 1$, n is replaced with $n - c^k$, and nothing else is changed. Since it was possible to make change for n with the original instance, it should be clear that it is possible to make change for $n - c^k$ with the new instance. Furthermore, it follows by the inductive hypothesis that every solution for the new instance should use the value

$$\min(\lceil \tfrac{n-c^k}{c^k} \rceil, m_k - 1) = \min(\lceil \tfrac{n}{c^k} \rceil, m_k) = 1$$

for "s_k." On the other hand, it should be clear that, once you've started to make change for n by choosing a coin with denomination c^l, you must continue by making change for $n - c^l$ in an optimal way (with one fewer coin with denomination c^l available), and *this* implies the total number of coins with denomination c^l used must now be

$$1 + (\min(\lceil \tfrac{n}{c^k} \rceil, m_k) + 1) = \min(\lceil \tfrac{n}{c^k} \rceil, m_k),$$

as required to complete the proof. $\square$

If you're making change for n using

$$s_k = \min(\lceil \tfrac{n}{c^k} \rceil, m_k)$$

coins with denomination c^k (and no coins with higher denomination are available) then it should be clear that you must continue by making change for the amount $n - s_k c^k$ using the available coins with denominations $1, c, c^2, \dots, c^{k-1}$. This observation, and the previous claim, imply that the next one can be established by induction on k; its proof is left as an easy exercise.

Claim 3.4. *Let c, k, $m_0, m_1, \dots, m_k$, n be an instance of the coin changing problem such that it is possible to make change for n. Then the only solution for this instance of the problem is the sequence $s_0, s_1, \dots, s_k$ defined by the equations*

$$n_{k+1} = n,$$

and

$$s_i = \min(\lceil \tfrac{n_{i+1}}{c^i} \rceil, m_i) \quad \textit{and} \quad n_i = n_{i+1} - s_i c^i$$

for $i = k, k-1, \dots, 1, 0$. Furthermore, if $n_{k+1}, n_k, n_{k-1}, \dots, n_1, n_0$ are as defined by these equations then $n_0 = 0$.

```
$n_{k+1} := n$
**for** $i := k$ **down to** $0$ **do**
    $s_i := \min(\lceil \frac{n_{i+1}}{c^i} \rceil, m_i)$
    $n_i := n_{i+1} - s_i c^i$
**end for**
**if** $n_0 \neq 0$ **then**
    **for** $i := 0$ **to** $k$ **do**
        $s_i := 0$
    **end for**
**end if**
**return** $s_0, s_1, \ldots, s_k$
```

Figure 1: An Iterative Algorithm to Make Change When Coins are Limited

An algorithm for making change whose correctness is established by this claim is shown in Figure 1. It should be clear by inspection of this algorithm that it uses $O(k)$ arithmetic operations in the worst case.

(b) When I set this question, I had a solution for the second problem in mind in which a "greedy choice" for non-base instances was simply a coin of the smallest denomination that you could legally choose. In other words, if you were given an instance c, k, $m_0, m_1, \ldots, m_k$, and n for the problem, such that it was possible to make change for n, then a greedy choice would be a coin with denomination c^i, where i was chosen to be as small as possible such that $m_i > 0$ and that it was also possible to make change for $n - c^i$ (after the value of m_i was decremented by one). You could use a solution for the first version of the problem to decide whether it was possible to make change for n at all (setting $s_i = 0$ for all i if it wasn't), and you could use this to select the greedy choice, too.

As it happens, though, there is a *much* better solution for this last version of the problem than that!

Claim 3.5. *Let c, k, $m_0, m_1, \ldots, m_k$, n be an instance of the coin changing problem in which you wish to make change for n using* as many *coins as you can.*

Let

$$N = \sum_{i=0}^{k} m_i c^i,$$

the total value of all the coins available to you.

It is impossible to make change for n if $n > N$. Otherwise, $n \leq N$ and it is possible to make change for n if and only if it is possible to make change for $N - n$.

Furthermore, suppose that $\hat{s}_0, \hat{s}_1, \ldots, \hat{s}_k$ is a solution for an instance of the previous *version of the problem with inputs c, k, $m_0, m_1, \ldots, m_k$ and $N - n$, so that $0 \leq \hat{s}_i \leq m_i$ for $0 \leq i \leq k$,*

$$\sum_{i=0}^{k} \hat{s}_i c^i = N - n,$$

and

$$\sum_{i=0}^{k} \hat{s}_i$$

is as small as possible. Then the (only) solution for the new version of the problem, for the instance c, k, $m_0, m_1, \ldots, m_k$ and n, is the sequence $s_0, s_1, \ldots, s_k$, where

$$s_i = m_i - \hat{s}_i$$

for $0 \le i \le n$. That is, $0 \le s_i \le m_i$ for $0 \le i \le k$,

$$\sum_{i=0}^{k} s_i c^i = n,$$

and

$$\sum_{i=0}^{k} s_i$$

is as large as possible.

In other words, you should make change for n using the *largest* number of coins possible by computing the total value N of all the coins available, figuring out how to make change for $N - n$ using the *smallest* number of available coins that you can, and using this to decide which coins you *should not* include!

The proof of this claim is also left as an exercise. Note that it implies that the last version of this problem can be solved *substantially* more efficiently than the questions suggested!

In particular, this would also answer Question #4 as well as #3.